Probabilistic Model–Checking of Quantum Protocols

Simon Gay[1], Rajagopal Nagarajan[2,⋆], and Nikolaos Papanikolaou[2,⋆⋆]

[1] Department of Computing Science, University of Glasgow
simon@dcs.gla.ac.uk
[2] Department of Computer Science, University of Warwick
{biju,nikos}@dcs.warwick.ac.uk

Abstract. We establish fundamental and general techniques for formal verification of quantum protocols. Quantum protocols are novel communication schemes involving the use of quantum-mechanical phenomena for representation, storage and transmission of data. As opposed to quantum computers, quantum communication systems can and have been implemented using present-day technology; therefore, the ability to model and analyse such systems rigorously is of primary importance.
While current analyses of quantum protocols use a traditional mathematical approach and require considerable understanding of the underlying physics, we argue that automated verification techniques provide an elegant alternative. We demonstrate these techniques through the use of PRISM, a probabilistic model-checking tool. Our approach is conceptually simpler than existing proofs, and allows us to disambiguate protocol definitions and assess their properties. It also facilitates detailed analyses of actual implemented systems. We illustrate our techniques by modelling a selection of quantum protocols (namely superdense coding, quantum teleportation, and quantum error correction) and verifying their basic correctness properties. Our results provide a foundation for further work on modelling and analysing larger systems such as those used for quantum cryptography, in which basic protocols are used as components.

1 Introduction

In the 1980s Richard Feynman, David Deutsch, Paul Benioff, and other scientists realized that quantum–mechanical phenomena can be exploited directly for the manipulation, storage and transmission of information. The discovery of quantum algorithms for prime factorization [20] and unstructured search [7], which outperformed the best classical algorithms for these tasks significantly, opened up new vistas for computer science and gave an initial thrust to the emerging field of quantum computation. To implement a quantum algorithm, however, a large scale quantum computer is necessary and such a device has yet to be built.

⋆ R. Nagarajan is supported by EPSRC grant GR/S34090 and the EU Sixth Framework Programme (Project SecoQC: *Development of a Global Network for Secure Communication based on Quantum Cryptography*).
⋆⋆ N. Papanikolaou is supported by a Postgraduate Fellowship by the Department of Computer Science, University of Warwick.

Research in *quantum information,* on the other hand, has shown that quantum effects can be harnessed to provide efficient and highly secure communication channels, which can be built using current technology. Entangled quantum states, superpositions and quantum measurement are among the characteristics of the subatomic world which nature puts at our disposal; these and related phenomena enable the development of novel techniques for computation and communication with no rival in classical computing and communication theory.

The focus in this paper is on communication protocols involving the use of such phenomena. Quantum protocols have particularly important applications in cryptography. Several quantum protocols have been proposed for cryptographic tasks such as oblivious transfer, bit commitment and key distribution [8,14]. The BB84 protocol for quantum key distribution [2,15], which allows two users to establish a common secret key using a single quantum channel, has been shown to be unconditionally secure against all attacks [12]. This degree of security has never been guaranteed by any classical cryptographic protocol, and the discovery has incited widespread interest in the properties of quantum protocols. Furthermore, practical quantum cryptographic devices are commercially available (e.g. from the companies Id Quantique and MagiQ).

The quantum teleportation protocol [3] and the superdense coding protocol [4] are both interesting and important examples of non–cryptographic quantum communication. Also, any attempt to implement a realistic quantum communication system, or indeed a quantum computer, must account for noise and quantum decoherence; these phenomena may be tackled through the use of quantum error correction protocols [21]. We will study these protocols in detail.

We argue that detailed, automated analyses of protocols such as these facilitate our understanding of complex quantum behaviour and enable us to construct valuable proofs of correctness. Such analyses are especially important to manufacturers of commercial devices based on such protocols; the actual security of commercial quantum cryptographic systems, for example, is worth an in–depth investigation. Communication protocols have always been under scrutiny by computer scientists, who have developed numerous techniques for analysing and testing them, including process algebras, formal specification languages and automated verification tools. Automated verification techniques, such as *model-checking* and *theorem proving,* are frequently targeted at protocols and have been used to detect faults and subtle bugs. For instance, the FDR model-checker allowed Gavin Lowe to uncover a flaw in the Needham–Schroeder security protocol [19].

Our approach is distinguished by the ability to incorporate system parameters in models and to vary them during verification. Also, while manual proofs of correctness have to be rewritten to account for variations in a protocol, the models we use may be adapted easily to different scenarios. Although current model-checkers were developed primarily for the analysis of classical systems, we have found ways of using them to model quantum behaviour. To account for the probabilism inherent in quantum systems, we have chosen to use a *probabilistic* model–checker, in particular, the PRISM tool developed at the University of Birmingham [17].

The structure of this paper is as follows. Section 2 provides necessary background on quantum theory and the PRISM model checker. In Section 3, we discuss the issues associated with modelling quantum protocols, and explain how they may be resolved through explicit state space generation. This leads to a presentation of specific quantum protocols, namely superdense coding, quantum teleportation, and the quantum bit-flip code. We explain how the techniques of Section 3 have enabled us to analyse and validate these protocols with PRISM. Section 5 sets the stage for future work. Finally, we conclude with the conviction that our techniques may be applied to more substantial systems, containing both classical and quantum components.

Previous and Related Work. Formal verification of quantum protocols was advocated by Nagarajan and Gay in [13], where they use the process calculus CCS to formally specify BB84. Papanikolaou's M.Sc. thesis [16] details a preliminary analysis of the BB84 quantum key distribution protocol using the PRISM and the SPIN model checkers.

Acknowledgement. The authors would like to thank Marta Kwiatkowska's group at the University of Birmingham, especially David Parker and Gethin Norman, for assistance with the PRISM tool.

2 Preliminaries

2.1 Basic Concepts of Quantum Computation

We briefly introduce those aspects of quantum theory relevant to quantum protocols. More detailed presentations can be found in [14] and [18].

A *quantum bit* or *qubit* is a physical system which has two basis states, conventionally written $|0\rangle$ and $|1\rangle$, corresponding to one-bit classical values. These could be, for example, spin states of a particle or polarization states of a photon, but we do not consider physical details. According to quantum theory, a general state of a quantum system is a *superposition* or linear combination of basis states. A qubit has state $\alpha|0\rangle + \beta|1\rangle$, where α and β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$; states which differ only by a (complex) scalar factor with modulus 1 are indistinguishable. States can be represented by column vectors: $\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \alpha|0\rangle + \beta|1\rangle$. Formally, a quantum state is a unit vector in a Hilbert space, i.e. a complex vector space equipped with an inner product satisfying certain axioms. In this paper we restrict attention to collections of qubits.

The basis $\{|0\rangle, |1\rangle\}$ is known as the *standard* basis. Other bases are sometimes of interest, especially the *diagonal* (or *dual*, or *Hadamard*) basis consisting of the vectors

$$|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad \text{and} \quad |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

Evolution of a closed quantum system can be described by a *unitary transformation*. If the state of a qubit is represented by a column vector then a unitary transformation U can be represented by a complex-valued matrix (u_{ij}) such that $U^{-1} = U^*$, where U^* is the conjugate-transpose of U (i.e. element ij of U^* is

$\bar{u}_{ji}$). U acts by matrix multiplication:

$$\begin{bmatrix}\alpha'\\ \beta'\end{bmatrix} = \begin{bmatrix}u_{00} & u_{01}\\ u_{10} & u_{11}\end{bmatrix}\begin{bmatrix}\alpha\\ \beta\end{bmatrix}$$

A unitary transformation can also be defined by its effect on basis states, which is extended linearly to the whole space. For example, the *Hadamard* operator is defined by

$$|0\rangle \mapsto |+\rangle = \tfrac{1}{\sqrt{2}}|0\rangle + \tfrac{1}{\sqrt{2}}|1\rangle \qquad |1\rangle \mapsto |-\rangle = \tfrac{1}{\sqrt{2}}|0\rangle - \tfrac{1}{\sqrt{2}}|1\rangle$$

which corresponds to the matrix $\mathsf{H} = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}$. The *Pauli* operators, denoted by $\sigma_0, \sigma_1, \sigma_2, \sigma_3$, are defined by

$$\sigma_0 = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \quad \sigma_1 = \begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix} \quad \sigma_2 = \begin{bmatrix}0 & -i\\ i & 0\end{bmatrix} \quad \sigma_3 = \begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}$$

Measurement plays a key role in quantum physics. If a qubit is in state $\alpha|0\rangle + \beta|1\rangle$ then measuring its value gives the result 0 with probability $|\alpha|^2$ (leaving it in state $|0\rangle$) and the result 1 with probability $|\beta|^2$ (leaving it in state $|1\rangle$). For example, if a qubit is in state $|+\rangle$ then a measurement (with respect to the standard basis) gives result 0 (and state $|0\rangle$) with probability $\frac{1}{2}$, and result 1 (and state $|1\rangle$) with probability $\frac{1}{2}$. If a qubit is in state $|0\rangle$ then a measurement gives result 0 (and state $|0\rangle$) with probability 1.

To go beyond single-qubit systems, we consider tensor products of spaces (in contrast to the cartesian products used in classical systems). If spaces U and V have bases $\{u_i\}$ and $\{v_j\}$ then $U \otimes V$ has basis $\{u_i \otimes v_j\}$. In particular, a system consisting of n qubits has a 2^n-dimensional space whose standard basis is $|00\ldots0\rangle \ldots |11\ldots1\rangle$. We can now consider measurements of single qubits or collective measurements of multiple qubits. For example, a 2-qubit system has basis $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ and a general state is $\alpha|00\rangle + \beta|01\rangle + \gamma|10\rangle + \delta|11\rangle$ with $|\alpha|^2 + |\beta|^2 + |\gamma|^2 + |\delta|^2 = 1$. Measuring the first qubit gives result 0 with probability $|\alpha|^2 + |\beta|^2$ (leaving the system in state $\frac{1}{\sqrt{|\alpha|^2+|\beta|^2}}(\alpha|00\rangle + \beta|01\rangle)$) and result 1 with probability $|\gamma|^2 + |\delta|^2$ (leaving the system in state $\frac{1}{\sqrt{|\gamma|^2+|\delta|^2}}(\gamma|10\rangle + \delta|11\rangle)$); in each case we renormalize the state by multiplying by a suitable scalar factor. Measuring both qubits simultaneously gives result 0 with probability $|\alpha|^2$ (leaving the system in state $|00\rangle$), result 1 with probability $|\beta|^2$ (leaving the system in state $|01\rangle$) and so on; the association of basis states $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ with results $0, 1, 2, 3$ is just a conventional choice. The power of quantum computing, in an algorithmic sense, results from calculating with superpositions of states; all of the states in the superposition are transformed simultaneously (*quantum parallelism*) and the effect increases exponentially with the dimension of the state space. The challenge in quantum algorithm design is to make measurements which enable this parallelism to be exploited; in general this is very difficult.

The *controlled not* (CNot) operator on pairs of qubits performs the mapping $|00\rangle \mapsto |00\rangle$, $|01\rangle \mapsto |01\rangle$, $|10\rangle \mapsto |11\rangle$, $|11\rangle \mapsto |10\rangle$, which can be understood as

inverting the second qubit (the *target*) if and only if the first qubit (the *control*) is set. The action on general states is obtained by linearity.

Systems of two or more qubits may be in *entangled* states, meaning that the states of the qubits are correlated. For example, consider a measurement of the first qubit of the state $\frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$. The result is 0 (and the resulting state is $|00\rangle$) with probability $\frac{1}{2}$, or 1 (and the resulting state is $|11\rangle$) with probability $\frac{1}{2}$. In either case, a subsequent measurement of the second qubit gives a definite, non–probabilistic result which is identical to the result of the first measurement. This is true even if the entangled qubits are physically separated. Entanglement illustrates the key difference between the use of the tensor product (in quantum systems) and the cartesian product (in classical systems): an entangled state of two qubits is one which cannot be expressed as a tensor product of single-qubit states. The Hadamard and CNot operators can be combined to create entangled states: $\mathsf{CNot}((\mathsf{H} \otimes I)|00\rangle) = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$.

2.2 Probabilistic Model–Checking

PRISM is an acronym for *probabilistic symbolic model checker,* and is designed for modelling and validating systems which exhibit probabilistic behaviour. Whereas a logical model–checker, such as SPIN [9], only states whether a system model σ satisfies a temporal formula Φ, a tool such as PRISM computes the probability with which such a formula is satisfied, i.e. the value of $\mathrm{P}_{\sigma,\Phi} = \Pr\{\sigma \models \Phi\}$ for given σ and Φ. The models catered for by PRISM may incorporate specific probabilities for various behaviors and so may the formulas used for verification. Probabilistic models and PRISM–like tools find applications in numerous areas of computer science where random behaviour is involved. Oft–cited applications are randomized algorithms, real–time systems and Monte Carlo simulation. The application of probabilistic model–checking to quantum systems is entirely appropriate, since quantum phenomena are inherently described by random processes; to reason about such phenomena one must account for this.

PRISM uses a built–in specification language based on Alur and Henzinger's REACTIVE MODULES formalism (see [10,17] for details). Using this language the user can describe probabilistic behaviour. Internally, a PRISM model is represented by a *probabilistic transition system.* In such a system, each step in a computation is represented by a *move,* or *transition,* from a particular state s to a distribution π of successor states. For technical details, refer to [17].

The probabilistic temporal logic PCTL [5] is used as the principal means for defining properties of systems modelled in PRISM. It suffices for our purposes to remind the reader of the meaning of the operator $\mathcal{U}$, known as "unbounded until". The formula $\Phi_1 \,\mathcal{U}\, \Phi_2$ expresses the fact that Φ_1 holds continuously from the current state onward, *until eventually* Φ_2 becomes **true**. The PRISM property $\mathrm{P} \geqslant 1[\Phi_1 \,\mathcal{U}\, \Phi_2]$ states that the formula $\Phi_1 \,\mathcal{U}\, \Phi_2$ is true with certainty, i.e. with a probability of unity; we use PRISM to check whether such a property holds in a given model.

3 Fundamental Techniques

In order to use a classical probabilistic model–checker to verify quantum protocols, we need to model the quantum states that arise in a given protocol, and the effect of specific quantum operations on these states. PRISM itself only allows positive integer and boolean variables to be used in models. So how can we model the states of quantum systems, and the quantum operations arising in protocols, using only classical data types and arithmetic?

Single qubits can be in a superposition of two states, while classical variables can only take on a single value in any given state. The coefficients of these states can be any two complex numbers whose moduli squared sum to unity, and there is an uncountable infinity of these; of course, PRISM can only work with a finite state space. Furthermore, quantum systems consisting of many qubits can be in entangled states, which, unlike classical systems, cannot be decomposed into products of individual states. What is needed, therefore, is a means of representing quantum states fully and consistently, in a form that PRISM can handle.

Of all the possible quantum states of an n–qubit system, we identify the finite set of states which arise by applying the operations CNot, Hadamard (H), and $\sigma_0, \sigma_1, \sigma_2, \sigma_3$ to input states. We confine our analyses to protocols that involve *only* this restricted set of operations. At present, determining which states belong to this set is done manually, but we are considering ways of automating this.

Consider a very simple system: a single qubit, being acted upon through the Hadamard gate and through measurement in the standard basis. For our purposes, the state of the qubit may be $|0\rangle, |1\rangle$, or an equal superposition of the two. In fact, these states are sufficient to model the BB84 protocol for quantum key distribution [2]. The quantum states which we need to represent in order to model this simple system are thus:

$$|0\rangle, |1\rangle, \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \text{ and } \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$$

To model this small, finite set of quantum states, which is closed under the operation of the Hadamard gate and the Pauli operators, we represent each state by assigning a unique integer from 0 to 3 to it, and we use straightforward transitions from one integer value to another to model the action of the gate. The actual PRISM model for this, as well as the possible results of measurement, is listed in the appendix.

A protocol such as superdense coding, which we will discuss in Section 4.1, can be expressed as a step-by-step interaction with a *two–qubit system.* In order to model the states of 2– and 3–qubit systems, the quantum operators and the measurements which arise in this and related protocols such as teleportation, we have developed a code generation tool called PRISMGEN. This tool generates a PRISM code fragment, or *module,* in which each quantum state is represented by a unique positive integer. Every quantum operator used in a particular protocol is coded as a set of deterministic transitions from one quantum state to another. PRISMGEN calculates these transitions by multiplying the unitary matrix, which corresponds to a particular operator, with each quantum state vector of interest.

A measurement is modelled by a set of probabilistic transitions, leading to the various possible outcomes with equal probability. For simplicity, we have only considered states whose measurement outcomes are all equiprobable, although PRISM does allow us to model the more general case.

From the overall state space for a two–qubit system, a certain subset is closed under the CNot, Hadamard and Pauli operations. This subset consists of the following states, which are 24 in total:

- 4 states corresponding to the four basis vectors, i.e. $|00\rangle, |01\rangle, |10\rangle, |11\rangle$.
- 12 states which are sums of two basis vectors, i.e. of the form $\frac{1}{\sqrt{2}}|xy\rangle \pm \frac{1}{\sqrt{2}}|x'y'\rangle$ with $x \neq x', y \neq y'$ and $x, x', y, y' \in \{0, 1\}$.
- 8 states which are sums of all four basis vectors.

Proposition. The above set of 24 states is closed under the CNot, Hadamard and Pauli operations.

Proof. These states can be expressed in the following way.

1. The single basis vectors: $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$
2. The states containing two basis vectors can be separated into three subclasses:
   (a) $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle) \otimes |0\rangle$, $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle) \otimes |1\rangle$
   (b) $|0\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$, $|1\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$
   (c) $\frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle)$, $\frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle)$
3. The states containing four basis vectors can be expressed in any of the forms:
   (a) $\frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle) \otimes |0\rangle \pm \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle) \otimes |1\rangle$
   (b) $|0\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle) \pm |1\rangle \otimes \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$
   (c) $\frac{1}{2}(|00\rangle \pm |01\rangle \pm |10\rangle \pm |11\rangle)$

It is obvious that each set (1.)–(3.) individually is closed under each σ_i (applied to either qubit) and CNot these operations are permutations of the basis vectors. Each set has an evident symmetry among the basis vectors (taking (3.a) and expanding (2.) into an explicit list of states). Applying H to the first qubit gives a bijection between (1.) and (2.a), between (2.b) and (3.a), and between (2.c) and (3.b). Applying H to the second qubit is similar. □

Our PRISMGEN tool enumerates these states and calculates the transitions corresponding to the various operations. The resulting PRISM module can be included as part of any model which involves measurements and the application of these operations to a system of two qubits.

The situation with a system of three qubits is similar. We have developed a 3–qubit version of PRISMGEN, which gives us the ability to model protocols such as those for quantum teleportation and quantum error correction.

4 Illustrative Examples

We have been able to model a certain number of quantum protocols using the aforementioned techniques. These include: (1) superdense coding, which is a procedure for encoding pairs of classical bits into single qubits; (2) quantum teleportation, which allows the transmission of a quantum state without the use of an intervening quantum channel; and (3) quantum error correction, namely the qubit flip code, which corrects a single bit flip error during transmission of quantum bits. The source files for the models in this section will be made available online at `http://go.warwick.ac.uk/nikos/research/`.

4.1 Superdense Coding

The simplest quantum protocol which we will use to illustrate our techniques is the superdense coding scheme [4]. This scheme makes it possible to encode a pair of classical bits on a single qubit. With superdense coding, a quantum channel with a capacity of a single qubit is all that is necessary to transmit twice as many bits as a serial classical channel. Superdense coding is essentially a computation on a two–qubit system; therefore, the PRISM model of this protocol uses the 2–qubit version of PRISMGEN. We begin with a description of the protocol, and proceed to show how it is modelled and verified with PRISM.

The setting for superdense coding involves two parties, conventionally named Alice and Bob, who are linked by a quantum channel and share a pair of entangled qubits. The objective is for Alice to communicate the binary number xy — henceforth termed the *message* and denoted by (x, y), with $x, y \in \{0, 1\}$ — by transmitting a single qubit to Bob. The superdense protocol takes advantage of the correlations between qubits P_1 and P_2, which are in an entangled quantum state. Alice essentially influences this state in such a way that Bob's measurement outcome matches the message of her choice. The quantum circuit diagram for the superdense coding procedure is shown in Fig. 1.

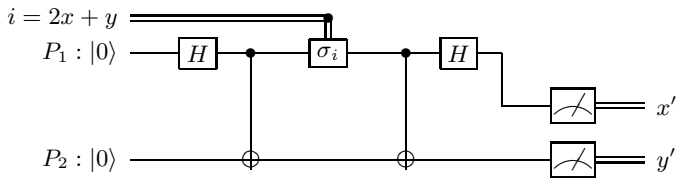

Fig. 1. Quantum circuit diagram for the superdense coding protocol.

Quantum circuit diagrams are a convenient means for expressing computations on qubits; while these are mostly self–explanatory, the reader is referred to standard texts [8,14] for explanations of the notation. For clarity, we describe the superdense coding in full below.

1. Two qubits, P_1 and P_2, are placed in an entangled state using the Hadamard and CNot operations. Alice is given P_1, and Bob is given P_2.
2. Alice selects a message, (x, y), and applies the ith Pauli operator, σ_i, to P_1, where $i = y + x(2 + (-1)^y)$. She transmits this particle to Bob.

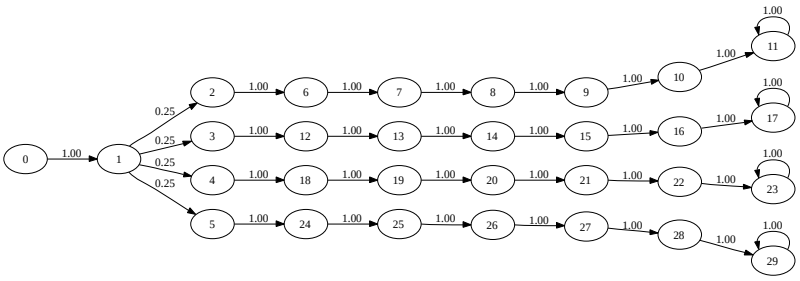

Fig. 2. Internal probabilistic state transition system corresponding to the PRISM model of superdense coding.

3. Bob applies the CNot gate from P_1 to P_2, and then he applies the Hadamard gate to the former.
4. Bob measures the two particles, thus obtaining a pair of classical bits, (x', y'). If no disturbance has occurred, this pair of bits will match the original message, i.e. $(x', y') = (x, y)$.

In what follows, we discuss the salient aspects of the PRISM model for this protocol, which is listed in the appendix; this will serve as a simple demonstration of the nature and structure of similar PRISM models. The model of superdense coding consists of four PRISM modules. Of these four, one module is generated automatically by PRISMGEN and describes the possible states of the two qubits. There is a module specifying Alice's actions, and similarly one for Bob's. The actions in these two modules correspond exactly to the successive application of quantum gates in Fig. 1.

Before we examine the workings of this model in detail, consider the following observations, which highlight the capabilities of PRISM. In the PRISM model, Alice's first action is to select one of the four possible messages (represented by the integers 0, 1, 2, 3); each message has an equal probability, $\frac{1}{4}$, of being chosen. This is an assumption we made when constructing this model, but it is possible to specify different respective probabilities for the four choices. Another point worth noting is that, depending on which message is chosen, the protocol proceeds in one of four distinct ways; PRISM actually considers *all* these possibilities when testing the validity of a property. This is precisely why we advocate the use of model-checking for these analyses, as opposed to simulation of quantum protocols, proposed elsewhere; simulators only treat one of several possible executions at a time.

PRISM interprets the superdense coding model as the probabilistic transition system[1] depicted in Fig. 2. The nodes in the graph correspond to the internal state numbers which PRISM assigns to each step in the protocol. Each internal state number corresponds to a tuple with the states of all variables in a particular model.

[1] Note that the probabilities in this diagram arise from Alice's choice of message, not from measurement outcomes. In general, it is measurement that produces probabilistic transitions.

State number	`alice_step`	`msg`	`bob_step`	`result`	`state`	Quantum State
0	0	0	0	0	0	$\|00\rangle$
1	1	0	0	0	8	$\frac{1}{\sqrt{2}}(\|00\rangle + \|11\rangle)$
4	2	2	0	0	8	$\frac{1}{\sqrt{2}}(\|00\rangle + \|11\rangle)$
18	3	2	0	0	9	$\frac{1}{\sqrt{2}}(\|01\rangle + \|10\rangle)$
19	3	2	1	0	9	$\frac{1}{\sqrt{2}}(\|01\rangle + \|10\rangle)$
20	3	2	2	0	7	$\frac{1}{\sqrt{2}}(\|00\rangle - \|10\rangle)$
21	3	2	3	0	2	$\|10\rangle$
22	3	2	4	0	2	$\|10\rangle$
23	3	2	5	2	2	$\|10\rangle$

Table 1. The transitions of the PRISM model for superdense coding for the case when the message chosen by Alice is **(1,0)**.

Read from left to right, Fig. 2 shows the succession of internal state numbers for the four possible messages that Alice may send to Bob in superdense coding, The initial state, labelled 0, is where all variables are first set. In internal state 1, Alice makes her choice of message, setting the `msg` variable accordingly and leading to one of the internal states 2, 3, 4, or 5 with equal probability. The succession of PRISM's internal states in Table 1, which includes the value of each variable in the model, corresponds to the case in which Alice has chosen the message **(1, 0)** and, hence, applied the σ_1 operator to her qubit. The quantum state of the two-qubit system is represented by the variable `state`, which corresponds to the actual quantum state indicated in the final column of this table.

When Bob has finished his measurement, and the dense coding protocol terminates, one of the internal states 11, 17, 23, 29 is reached, corresponding to the final nodes in the graph in Fig. 2. The property required for verification must be expressed in terms of the final state. When the dense coding protocol terminates, Bob's measurement result, i.e. the pair of classical bits (x', y'), must match Alice's original choice (x, y). This requirement is expressed using PCTL, as follows:

$$\mathrm{P} \geqslant 1 \; [\; \mathbf{true}\; \mathcal{U}\; ((\texttt{protocol\_finished}) \;\wedge\; (\texttt{result} = \texttt{msg}))\;] \qquad (1)$$

The PCTL formula in (1) stipulates that the probability of Bob's result matching Alice's choice is 1. Model–checking with PRISM confirms that this property holds (i.e. this property is **true** for all executions of the model). We have thus proven, using the PRISM model–checker, that the dense coding protocol always succeeds in transmitting two classical bits using a single qubit. Clearly, this is not difficult to prove by hand; however, we have used dense coding as a simple demonstration of our approach.

4.2 Quantum Teleportation

Our next example is the quantum teleportation protocol [3], which involves a computation on three qubits. Teleportation is a process that exploits entanglement in order to transmit an arbitrary qubit state using only a classical communications channel. The quantum circuit for teleportation is shown in Fig.3.

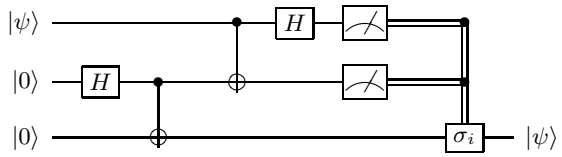

Fig. 3. Quantum circuit diagram for the teleportation protocol.

The PRISM model of teleportation is similar in appearance to that for superdense coding, and it is not included here due to lack of space. It is a transformation on a collection of three qubits, as opposed to the two for superdense coding. This calls for the 3-qubit version of PRISMGEN. Other than this, the PRISM model itself is unremarkable, and matches the structure of the quantum circuit for teleportation, given in the appendix. Verifying the teleportation protocol with PRISM is more involved. Short of manual calculation, it is not possible to predict what the quantum state of the entire 3-qubit system will be at the end of the teleportation protocol; indeed, there are several possible final states, depending on which quantum state Alice chooses to transmit to start with. We are interested in checking that the state of Bob's qubit matches Alice's original qubit state, $|\psi\rangle$, which is assumed to be one of $|0\rangle, |1\rangle, |+\rangle |-\rangle$. To formulate a usable property for verification, we need to express this requirement in terms of the overall state of the 3–qubit system.

Formally, the specification of the teleportation protocol is this: if the initial state of the 3-qubit system is of the form $|\psi\rangle \otimes |00\rangle$, then the final state will be of the form $|\phi\rangle \otimes |\psi\rangle$, where $|\phi\rangle$ is a two–qubit state. Let's consider this in more detail. If Alice chooses to teleport $|\psi\rangle = |0\rangle$, the final state of the 3–qubit system will be of the form $|\phi\rangle \otimes |0\rangle$. Similarly, if Alice chooses to teleport $|\psi\rangle = |1\rangle$, the final state of all three qubits will be of the form $|\phi\rangle \otimes |1\rangle$. Finally, if Alice chooses to teleport the superposition $|\psi\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$, the final state of the three qubits will be of the form $|\phi\rangle \otimes \left(\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle\right)$.

Clearly, the PCTL property necessary for verification will depend on the choice of $|\psi\rangle$; it will stipulate that, when the teleportation protocol has completed, the final state of the 3-qubit system will have one of the forms given above. In particular, if the input state is $|0\rangle$, the necessary property is

$$\mathrm{P} \geqslant 1 \; [\; \mathbf{true}\; \mathcal{U}\; ((\texttt{telep\_end}) \;\wedge\; ((\texttt{st} = s_1) \;\vee\; \cdots \;\vee\; (\texttt{st} = s_n)))\;] \qquad (2)$$

where `telep_end` is a predicate which is `true` when the protocol completes, and the values $s_1, \ldots, s_n$ represent quantum states of the form $|\phi\rangle \otimes |0\rangle$. If the input state is $|1\rangle$, the necessary property has exactly the same form as (2), but the values $s_1, \ldots, s_n$ represent quantum states of the form $|\phi\rangle \otimes |1\rangle$; similarly for the case when the input state is the superposition $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$.

In other words, in order to formulate the property needed to verify the protocol, we need to choose the input states and determine the possible final states of the three–qubit system in advance. This may be seen as begging the question; there is little point in verifying a protocol whose final outcome has already been calculated by hand. We have developed an auxiliary tool to PRISMGEN,

which computes the internal state numbers $s_1, \ldots, s_n$ corresponding to the desired final states. When the PCTL property for a particular input is supplied to PRISM, the tool proves that the teleportation model works as expected. Since the model–checker necessarily constructs a finite state space for the model, the teleportation protocol can only be verified for a specific, known set of inputs, rather than an arbitrary quantum state.

4.3 Quantum Error Correction

Our third and final example is the quantum bit–flip code for error correction [21]. In order to correct a single bit flip error, which may occur during the transmission of a single qubit state, this code represents the state by using a collection of three qubits. In particular, the qubit state $|0\rangle$ is encoded as $|000\rangle$ and the state $|1\rangle$ is encoded as $|111\rangle$. A bit flip error on the second qubit, for example, transforms $|000\rangle$ into $|010\rangle$.

In order to detect such an error, two additional qubits are used; they are known as *ancillas*. By applying a sequence of operations and measurements to the ancillas, the so–called *error syndrome* is obtained, which determines the location of the error. Then, the σ_1 operator is applied to the erroneous qubit, thus restoring the initial quantum state of the 3–qubit system. The quantum circuit for the bit–flip code is given in Fig. 4.

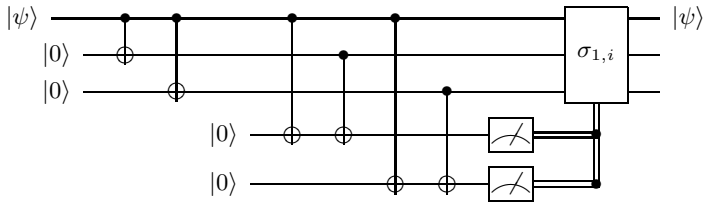

Fig. 4. Quantum circuit diagram for the qubit bit–flip code.

For the diagram we have assumed that a bit–flip error does occur prior to the computation of the syndrome.

Our PRISM model of the protocol for the quantum bit–flip code includes a channel which perturbs the transmitted qubit with a chosen probability; this probability is a parameter of the model, and can be varied as required. The model uses the output from the 3–qubit version of the PRISMGEN tool. When the syndrome computation is taken into account, there are in total five qubits whose states need to be modelled; since we have not yet implemented a code generator for 5–qubit quantum systems, the state transitions for the syndrome computation are calculated in advance and manually coded into PRISM.

To verify the correctness of the quantum bit–flip code, we need to ensure that: independently of which of the three qubits is perturbed and with what probability this occurs, the protocol does succeed in correcting the error. Thus, at the end of the protocol, the state of the 3–qubit system should be in one of the following forms (where $|\phi\rangle$ is a two–qubit state):

$$|0\rangle \otimes |\phi\rangle\,, \text{ if the input state was } |0\rangle \tag{3}$$

$$|1\rangle \otimes |\phi\rangle\,, \text{ if the input state was } |1\rangle \tag{4}$$

$$\left(\frac{1}{\sqrt{2}}\,|0\rangle + \frac{1}{\sqrt{2}}\,|1\rangle\right) \otimes |\phi\rangle\,, \text{ if the input state was } \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \tag{5}$$

The properties used in PRISM to verify the protocol are analogous to those for teleportation, taking the form

$$\mathrm{P} \geqslant 1\ [\ \mathbf{true}\ \mathcal{U}\ ((\texttt{qbf\_end})\ \wedge\ ((\texttt{st} = s_1)\ \vee\ \cdots\ \vee\ (\texttt{st} = s_n)))\ ] \tag{6}$$

where `qbf_end` is a predicate which holds when the protocol completes, and the values $s_1, \ldots, s_n$ represent quantum states of one of the forms given in (3)–(5). PRISM confirms that the protocol does indeed leave the 3–qubit system in one of these forms, depending on the input, as expected.

5 Challenges and Future Prospects

We have demonstrated our approach to the analysis of quantum communication protocols using three simple examples. There is significant scope for future work, ranging from improvements to our current code–generation techniques, to the automated verification of larger systems, such as quantum cryptographic devices.

State–Space Construction Techniques. At present we explicitly construct state spaces and transition tables for systems involving up to three qubits and the H, CNot and σ_i operators. We have informally reached the conclusion that, for any number of qubits, there is a finite set of states which is closed under these operators. It is not directly obvious how many states these are, but this could be established computationally. There is a mathematical framework called the stabilizer formalism , which could be used to calculate these states. Stabilizer circuits, which only include Clifford group gates (Hadamard, CNot and the *phase gate* [8,14]) and single–qubit measurements, are important in quantum error correction and fault-tolerant computation. Furthermore, the Gottesman–Knill theorem states that stabilizer circuits can be efficiently simulated by a classical computer [1]. We conjecture that there is a close correspondence between the Gottesman–Knill Theorem and the calculation of the closed set of states outlined here (note that the Pauli operators are derivable from the Clifford group gates). It is perhaps not surprising that there should be a connection between a class of quantum systems which can be efficiently simulated by a classical computer, and the class of quantum systems which can be effectively model-checked.

We are also investigating another approach, based on a direct representation of the coefficients in a quantum state as PRISM variables. By not normalizing the states in the conventional way, the need for real numbers can be avoided: for example, the state $\frac{\sqrt{3}}{2}|0\rangle + \frac{1}{2}|1\rangle$ would be represented by storing integer multiples of the coefficients. The correct normalization factor would be applied when calculating the probabilities for the measurement transitions. By storing

coefficients and calculating the probabilities explicitly within a PRISM model, we would avoid the commitment in advance to a state space of a particular structure. Initial attempts to do this have indicated that it is not straightforward to implement with PRISM: to support general coefficients with a reasonable precision, the range of values of the PRISM variables must be large, and this leads to difficulties with generating the state space.

High–Level Modelling Languages. The guarded transitions of PRISM's modelling language make it awkward to express some basic control structures such as sequencing. Each PRISM module typically requires a variable which acts as a program counter and must be explicitly incremented in each transition. We intend to develop automatic translations from the high–level process calculus CQP [6] into PRISM's native language. Combining such a specification formalism for protocol models with a logic for defining properties will allow us to verify quantum protocols at a higher level. Ultimately we would like to make use of existing work on quantum logic, for example [11].

Modelling Larger Systems. Our ultimate aim is to construct models of larger systems which combine quantum and classical components, or which combine more than one quantum protocol. For example, we are working on augmenting an existing model [16] of the BB84 key–distribution protocol with descriptions of authentication, secret–key reconciliation, and privacy amplification protocols [8]. As PRISM allows probabilities of particular events to be calculated directly, we can obtain numerical values of probability, such as those that arise in mathematical analyses of security; we have taken advantage of this capability in our existing model of BB84. More complex protocols generally involve larger numbers of qubits, leading to ever greater state spaces for verification. However, it will often be the case that the quantum state of a large system can be separated into independent parts. By using a high–level modelling language such as CQP, we will be able to identify non–interacting parts of the quantum state by static analysis.

6 Conclusions

We have established, for the first time, techniques for analyzing and verifying quantum communication systems. Our key contributions are the development of a general approach to modelling the state space of systems of several qubits, and the introduction of techniques for defining properties of quantum protocols in the logic PCTL. We have illustrated our approach by modelling and verifying three example protocols (superdense coding, quantum teleportation, and quantum error correction) using PRISM. Although these examples are simple, they are important building blocks of the theory of quantum communication. Our approach is conceptually simpler than existing mathematical proofs, and allows us to disambiguate protocol definitions and assess their properties. What is more, it is easy to investigate the effect of varying specific parameters and details once a model of a protocol has been built. Having established fundamental and general techniques for formal verification of quantum protocols, we are in a strong

position to carry out end–to–end verifications of larger systems, such as those used for quantum cryptography.

References

1. Aaronson, S., and Gottesman, D. Improved simulation of stabilizer circuits. Available at arXiv.org. Record: quant-ph/0406196., 2003.
2. Bennett, C. H., and Brassard, G. Quantum cryptography: Public key distribution and coin tossing. In *Proceedings of International Conference on Computers, Systems and Signal Processing* (December 1984).
3. Bennett, C. H., Brassard, G., Crépeau, C., Jozsa, R., Peres, A., and Wootters, W. K. Teleporting an unknown quantum state via dual classical and Einstein–Podolsky–Rosen channels. *Physical Review Letters 70* (1993), 1895–1899.
4. Bennett, C. H., and Wiesner, S. J. Communication via one- and two-particle operators on Einstein–Podolsky–Rosen states. *Physical Review Letters 69*, 20 (1992), 2881—2884.
5. Ciesinski, F., and Grösser, M. On probabilistic computation tree logic. In *Validation of Stochastic Systems* (2004), pp. 147–188.
6. Gay, S., and Nagarajan, R. Communicating quantum processes. In *POPL '05: Proceedings of the 32nd ACM Symposium on Principles of Programming Languages, Long Beach, California* (January 2005).
7. Grover, L. K. A fast quantum mechanical algorithm for database search. In *Proc. 28th Annual ACM Symposium on the Theory of Computing (STOC)* (1996), pp. 212–219.
8. Gruska, J. *Quantum Computing*. McGraw–Hill International, 1999.
9. Holzmann, G. *The SPIN Model Checker: Primer and Reference Manual*. Pearson Education, 2003.
10. Kwiatkowska, M., Norman, G., and Parker, D. Modelling and verification of probabilistic systems. In *Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems*, P. Panangaden and F. V. Breugel, Eds. American Mathematical Society, 2004. Volume 23 of crm Monograph Series.
11. Mateus, P., and Sernadas, A. Reasoning about quantum systems. In *Proceedings of the Ninth European Conference on Logics in Artificial Intelligence (JELIA'04)* (2004), no. 3229 in Lecture Notes in Artificial Intelligence, Springer.
12. Mayers, D. Unconditional security in quantum cryptography. *Journal of the* acm *48*, 3 (2001), 351—406.
13. Nagarajan, R., and Gay, S. Formal verification of quantum protocols. Available at arXiv.org. Record: quant-ph/0203086, 2002.
14. Nielsen, M. A., and Chuang, I. L. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.
15. Papanikolaou, N. Introduction to quantum cryptography. *ACM Crossroads Magazine 11.3* (2005), 10—16.
16. Papanikolaou, N. Techniques for design and validation of quantum protocols. Master's thesis, Department of Computer Science, University of Warwick, 2005.
17. Parker, D., Norman, G., and Kwiatkowska, M. prism 2.0 users' guide, February 2004.
18. Rieffel, E., and Polak, W. An introduction to quantum computing for non–physicists. acm *Computing Surveys 32*, 3 (2000), 300—335.
19. Ryan, P., Schneider, S., Goldsmith, M., Lowe, G., and Roscoe, B. *Modelling and Analysis of Security Protocols*. Pearson Education, 2001.

20. Shor, P. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings of 35th Annual Symposium on Foundations of Computer Science* (1994), IEEE Press.
21. Steane, A. M. Quantum computing and error correction. In *Proceedings of the NATO Advanced Research Workshop* (Mykonos, 2000), A. Gonis and P. Turchi, Eds., IOS Press, pp. 284—298.

7 Appendix

7.1 PRISM Model of a Single Qubit

```
probabilistic
module Qubit
state : [0..3];  // 0 is |0>, 1 is |1>, 2 is |+>, 3 is |->
result : [0..1]; // Result of measurement in standard basis
[hadamard] (state=0) -> (state'=2);
[hadamard] (state=1) -> (state'=3);
[hadamard] (state=2) -> (state'=0);
[hadamard] (state=3) -> (state'=1);
[measure] (state=0) -> (state'=0)&(result'=0);
[measure] (state=1) -> (state'=1)&(result'=1);
[measure] (state=2) -> 0.5 : (state'=0)&(result'=0)
                     + 0.5 : (state'=1)&(result'=1);
[measure] (state=3) -> 0.5 : (state'=0)&(result'=0)
                     + 0.5 : (state'=1)&(result'=1);
endmodule
```

7.2 PRISM Model of Superdense Coding

```
probabilistic
//initialize to 1/sqrt2 (|00>+|11>), which is state=8
const int INITSTATE = 8;
formula AL_FIN = (alice_step=3);
formula BOB_WAIT = (bob_step=0);
module Alice
alice_step : [0..3];
msg : [0..3];
// 1="00", 2="01", 3="10", 4="11"
[initialize]   BOB_WAIT &(alice_step=0) -> (alice_step'=1);
[]             BOB_WAIT &(alice_step=1) ->
                          1/4 : (msg'=0) & (alice_step'=2)
                        + 1/4 : (msg'=1) & (alice_step'=2)
                        + 1/4 : (msg'=2) & (alice_step'=2)
                        + 1/4 : (msg'=3) & (alice_step'=2);
[sigma0] BOB_WAIT & (alice_step=2) & (msg=0) -> (alice_step'=3);
[sigma1] BOB_WAIT & (alice_step=2) & (msg=1) -> (alice_step'=3);
[sigma3] BOB_WAIT & (alice_step=2) & (msg=2) -> (alice_step'=3);
[sigma2] BOB_WAIT & (alice_step=2) & (msg=3) -> (alice_step'=3);
```

```
[sendtoBob] BOB_WAIT &(alice_step=3) -> (alice_step'=3);
endmodule

module Bob
bob_step : [0..5];
result : [0..3];
// 1="00", 2="01", 3="10", 4="11"
[sendtoBob]   AL_FIN & (bob_step=0) -> (bob_step'=1);
[cnot]        AL_FIN & (bob_step=1) -> (bob_step'=2);
[hadamard1]   AL_FIN & (bob_step=2) -> (bob_step'=3);
[measure]     AL_FIN & (bob_step=3) -> (bob_step'=4);
[]            AL_FIN & (bob_step=4) ->
                   (result'=outcome) & (bob_step'=5);
[]            AL_FIN & (bob_step=5) -> (bob_step'=5);
//End of protocol
endmodule
// ...
// Automatically generated module with sigma0,sigma1,...
// actions not included here due to its repetitivity
// and the lack of space.
```